Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

A Latching Recovery Funnel for Neural Swing-Up Controllers of N -Link Cart–Pendulums

SHASHANK KUMAR
Indian Institute of Technology Roorkee
Mayank Goswami
mayankfph@iitr.ac.in

Indian Institute of Technology Roorkee https://orcid.org/0000-0001-8829-9980

Research Article

Keywords: underactuated systems, inverted pen- dulum, swing-up control, neuroevolution, LQR, hybrid control

Posted Date: November 17th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11739298/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

A Latching Recovery Funnel for Neural Swing-Up Controllers of N-Link Cart–Pendulums

Shashank K[1] and Mayank Goswami,[1,2]

[1]Department of Physics, IIT Roorkee
[2]MF School of AI and DS, IIT Roorkee
mayank.goswami@ph.iitr.ac.in

***Abstract*—Neural swing-up controllers for multi-link cart–pendulums are usually optimized from a canonical initial condition, the chain hanging at rest. We show that such a controller, although it swings up and balances $N = 1$–4 links from rest, almost never recovers after a large disturbance knocks the chain down: over 200 random knock-down states per link count it recovers in 97% of trials for $N = 1$ but in at most 1% for $N \geq 2$. The cause is that the memoryless swing-up policy keeps injecting force into a chain whose state lies far from the trajectories it was trained on. We propose a *recovery funnel* built into the same feedforward network as a small, analytically constructed circuit: a downward LQR damper that brings the cart to the center and the links to rest, a detector neuron that fires when the state is at downward rest, and a single self-recurrent ReLU *latch* neuron that holds the swing-up branch active for a fixed time. The circuit uses only ReLU and sine units, keeps every weight below the original parameter bound, and leaves nominal behavior unchanged. In simulation it raises knock-down recovery to 200/200 for every $N \in \{1, \ldots, 4\}$ and recovers from all 24 extreme push and tip-yank scenarios, at the cost of longer recovery times (median 3.3 s for $N = 1$ to 12.6 s for $N = 4$). The funnel relies on exact convergence to rest in a noise-free simulator; we discuss this and other limitations. Code, models and the evaluation harness are open source.**

***Index Terms*—underactuated systems, inverted pendulum, swing-up control, neuroevolution, LQR, hybrid control**

I. Introduction

Swinging up and balancing a chain of N links on a cart is a standard benchmark for nonlinear control of underactuated systems and for learning-based control. Classical solutions combine an energy-based or trajectory-based swing-up law with a linear stabilizer near the upright equilibrium [1]–[4]. Neuroevolution and reinforcement learning methods learn the same behavior from reward [5]–[7].

A practical requirement that is easy to overlook is *reentry*: after a large external disturbance knocks the chain out of the upright region, the controller should bring it back up. For controllers trained mainly from the canonical hanging-at-rest initial condition, nothing in the objective rewards this. We observed this failure in an interactive simulator: a swing-up controller that worked reliably from rest would, after a hard push, thrash the cart against the track limits without ever raising the chain again.

This paper documents that failure and a fix. Our contributions are:

1) An open-source, real-time simulator for N-link cart–pendulums (Rust; native and WebAssembly) with interactive mouse perturbations and a gated neural controller whose upright branch is an LQR law written as network weights and whose swing-up branch is optimized by a genetic algorithm (Section III–IV).
2) A quantitative characterization of the re-entry failure of this baseline for $N = 1$–4 (Section VI).
3) A *recovery funnel*: a small circuit of ReLU and sine units, including one self-recurrent latch neuron, that is added to the trained network without retraining and that sequences *damp to rest* $\rightarrow$ *swing up* $\rightarrow$ *balance* (Section V).
4) An ablation on 800 random knock-down states, 200 random upright tilts and 24 extreme pushes comparing the network with and without the funnel (Section VI).

We stress at the outset that the funnel is *designed*, not learned: its weights are computed from a linearization and a few hand-chosen constants. The point of the paper is that a memoryless learned policy plus a tiny amount of structured memory solves a failure mode that the learned policy alone does not.

II. Related Work

Swing-up control. Energy shaping swings up the single pendulum robustly [1], [2]. For double and triple pendulums on a cart, successful designs rely on precomputed feedforward trajectories with tracking feedback [3], [4], which are valid only in the neighborhood of the planned trajectory and therefore require the system to start at (or be brought to) a known state.

Funnels and sequential composition. Composing controllers whose regions of attraction ("funnels") feed into one another is the idea of sequential composition [8] and of LQR-trees [9]. Our circuit is a minimal instance of this idea: a damping controller whose goal region is the entrance of the swing-up trajectory, followed by the swing-

TABLE I
PLANT PARAMETERS (IDENTICAL UNIFORM RODS FOR EVERY N).

Quantity	Symbol	Value
Cart mass / damping	M_c, b_c	1.5 kg, 0.1 N s/m
Link mass / length	m_i, L_i	0.5 kg, 0.6 m
Joint damping	b_i	0.005 N m s/rad
Gravity	g	9.81 m/s^2
Force limit	F	40 N
Track half-width	$x_{\max}$	2.5 m
Wall stiffness / damping	k_w, c_w	2000 N/m, 50 N s/m
Time step	Δt	2 ms (RK4)

up and upright controllers. Unlike those works we provide no formal verification of the regions.

Neuroevolution. NEAT [6] and CoSyNE [7] evolve networks for pole balancing. Our software includes a NEAT implementation, but the controllers studied here use a fixed, hand-structured topology whose free weights are evolved (Section IV).

III. System Model

A. Dynamics

The generalized coordinates are $q = [x, \theta_1, \ldots, \theta_N]^\top$, where x is the cart position and θ_i the absolute angle of link i from the upward vertical ($\theta_i = \pi$ is hanging down). The Euler–Lagrange equations are

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) + D\dot{q} = e_1\big(u + f_w(x,\dot{x})\big) + \tau_{\text{ext}}, \quad (1)$$

with mass matrix M, Coriolis matrix C, gravity vector G, viscous cart and relative-joint damping D, actuator force $u \in [-F, F]$ on the cart, and τ_{ext} the generalized force of an optional mouse "leash" (a spring–damper attached to any joint, Section VI-A). The track has compliant walls: $f_w = -k_w(|x| - x_{\max})\,\text{sign}(x) - c_w\dot{x}$ when $|x| > x_{\max}$ and the cart moves outward, otherwise 0. Equation (1) is solved for $\ddot{q}$ by Cholesky factorization and integrated with fourth-order Runge–Kutta at $\Delta t = 2$ ms; the controller is evaluated once per step (zero-order hold). Table I lists the parameters.

B. Observation

The controller sees the $2 + 3N$ features

$$o = \big[\tilde{x},\ \tilde{v},\ \sin\theta_1,\ \cos\theta_1,\ \tilde{\omega}_1,\ \ldots,\ \sin\theta_N,\ \cos\theta_N,\ \tilde{\omega}_N\big], \quad (2)$$

with $\tilde{x} = x/x_{\max}$ clipped to $[-2, 2]$, $\tilde{v} = \dot{x}/5$ and $\tilde{\omega}_i = \dot{\theta}_i/10$ clipped to $[-5, 5]$. The downward rest state maps to $o_\downarrow = [0, 0,\ 0, -1, 0,\ \ldots,\ 0, -1, 0]$.

IV. Baseline: Gated Neural Controller

A. Network formalism

A controller is a directed graph of neurons j with bias b_j, activation $\sigma_j \in \{\text{linear}, \tanh, \text{ReLU}, \text{sigmoid}, \sin\}$ and a depth coordinate $\ell_j \in [0, 1]$. An edge $i \to j$ with $i \neq j$ requires $\ell_i < \ell_j$, so evaluating neurons in order of ℓ is a single forward pass. In the funnel (Section V) we additionally allow a hidden neuron to have a *self-edge*, which reads the neuron's value from the previous time step:

$$v_j^t = \sigma_j\Big(b_j + \textstyle\sum_{i \neq j} w_{ij} v_i^t + w_{jj} v_j^{t-1}\Big). \quad (3)$$

The single output neuron is linear, and the applied force is $u = \text{sat}_F(F\,y)$. All weights and biases are bounded by $|w| \leq 300$.

B. Upright branch

Near the upright equilibrium we use discrete-time LQR on the zero-order-hold discretization of the linearization of (1) [10]. The diagonal Q, the scalar R, a gain scale and a cone threshold $c^\star$ are chosen by random search (600 candidates; 360 for $N = 4$) that maximizes the number of passed cases in a suite of upright perturbations (tilts, alternating "S"-shaped tilts, mouse tugs, cart kicks). Writing $\hat{z}$ for the state with $\sin\theta_i$ in place of θ_i, the law $u_{\text{up}} = -K\hat{z}$ is linear in the observation and is stored directly as weights.

C. Gates and push–pull routing

Two ReLU gates measure membership in the upright cone $\sum_i \cos\theta_i \geq Nc^\star$:

$$g_\downarrow = \text{ReLU}\big(s\,(Nc^\star - \textstyle\sum_i \cos\theta_i)\big), \quad (4)$$

$$g_\uparrow = \text{ReLU}\big(s\,(\textstyle\sum_i \cos\theta_i - N(c^\star - \delta))\big), \quad (5)$$

with slope $s = 150$ (260 for $N = 4$) and overlap $\delta = 0.02$ (0.01 for $N = 4$). Each signed command a is routed through a pair $\text{ReLU}(a - g)$ and $\text{ReLU}(-a - g)$ whose difference equals a when the inhibiting gate g is zero and vanishes when $g \geq |a|$. The baseline output is

$$y = \underbrace{\text{ReLU}\big(\tfrac{u_{\text{up}}}{F} - g_\downarrow\big) - \text{ReLU}\big(-\tfrac{u_{\text{up}}}{F} - g_\downarrow\big)}_{\text{upright}} + \underbrace{\text{ReLU}(1.25u_{\text{sw}} - g_\uparrow) - \text{ReLU}(-1.25u_{\text{sw}} - g_\uparrow)}_{\text{swing-up}}. \quad (6)$$

D. Swing-up branch

The swing-up command $u_{\text{sw}} \in [-1, 1]$ is the tanh output of a feed-forward subnetwork with link-dependent depth: one hidden layer of 2 units for $N = 1$ and up to four layers (6–4–2–1 units) for $N = 4$, mixing tanh, sin, ReLU and sigmoid units, with skip connections from every layer and from the inputs to the output. This gives 20, 61, 103 and 159 free parameters for $N = 1, \ldots, 4$.

They are optimized by a genetic algorithm with elitism (top 20%), uniform crossover, sparse and dense Gaussian mutation with a step size annealed from 0.35 to 0.04, and parallel rollouts on all CPU cores; population sizes and generations are 144/60, 192/90, 360/260 and 240/160 for $N = 1, \ldots, 4$. The fitness is a shaped per-step reward that favors high mean height, high $\min_i \cos\theta_i$, low angular velocity and a centered cart, plus a terminal bonus on the peak height; leaving the track ends the episode with a penalty. Episodes start at downward rest; for $N \leq 2$,

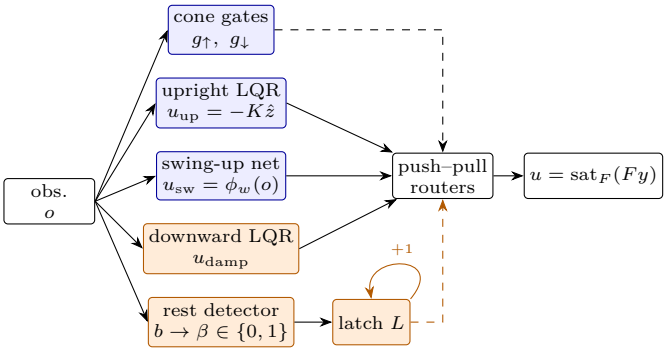


Fig. 1. Controller structure. Blue: baseline network (tag 0.2). Orange: recovery funnel (tag 0.3). Dashed lines are inhibitory gate signals; the latch L is the only recurrent unit.

candidates that already balance are also scored on one knocked-over and one perturbed hanging start, and mouse tugs are applied once upright.

E. Failure mode

Outside the upright cone, the baseline *always* applies u_{sw}. For $N \geq 2$ the swing-up is a finely timed sequence that works from (near) rest. After a large push the chain is swinging chaotically; the policy, which has no memory and has not seen such states, keeps injecting up to 40 N out of phase, and the chain is never at rest long enough for the learned sequence to begin. Section VI quantifies this.

V. The Recovery Funnel

The funnel adds three behaviors to the trained network without changing any existing weight: (i) *damp* the chain to downward rest when it is outside the upright cone and no swing-up is in progress; (ii) *detect* downward rest; (iii) *latch* the swing-up branch on for a fixed duration once rest is detected. Fig. 1 shows the structure.

A. Downward damper

We linearize (1) about the hanging equilibrium $\theta_i = \pi$, discretize with zero-order hold, and solve the discrete Riccati equation for the scaled pair $(\alpha A_d, \alpha B_d)$ with $\alpha > 1$. This places all closed-loop eigenvalues inside a disc of radius $1/\alpha$, i.e. it guarantees a decay rate of at least $\ln\alpha/\Delta t$ [10]. We use $Q = \text{diag}(45,\ 0.5\,\mathbf{1}_N,\ 20,\ 28{\cdot}1.3^{i-1})$, $R \in \{0.4, 0.8, 1.8, 3.0\}$ and $\alpha \in \{1.0015, 1.0015, 1.0018, 1.0018\}$ for $N = 1, \ldots, 4$, i.e. a decay rate of at least 0.75–0.90 s^{-1}, while keeping the force mostly within the 40 N limit. Because $\sin\theta_i \approx -(\theta_i - \pi)$ near the bottom, $u_{\text{damp}} = -K_\downarrow \delta z$ is again linear in o and is written as weights.

B. Rest detector

Three or more sine units compute $q_k = \sin(0.25\,o_k + \pi/2) = \cos(0.25\,o_k)$ for $o_k \in \{\tilde{x}, \tilde{v}, \tilde{\omega}_1, \ldots, \tilde{\omega}_N\}$. A ReLU unit then evaluates

$$b = \text{ReLU}\big(\rho - \Phi(o)\big), \qquad \rho = r_0^2/2, \tag{7}$$

$$\Phi(o) = 100(1 - q_{\tilde{x}}) + 40(1 - q_{\tilde{v}}) + \tfrac{1}{N}\textstyle\sum_i (1 + \cos\theta_i) + \tfrac{128}{N}\textstyle\sum_i (1 - q_{\tilde{\omega}_i}). \tag{8}$$

Because the clipped features satisfy $|0.25\,o_k| \leq 1.25 < \pi/2$, each term $1 - \cos(0.25\,o_k)$ is the Bregman divergence [11] of the strictly convex function $-\cos$ from 0; hence $\Phi \geq 0$ with equality only at $o_\downarrow$, and to second order Φ is a weighted squared distance to $o_\downarrow$. All of this is a single linear combination of existing inputs and sine units, so b is one neuron with bias $\rho - 269$ (the constant parts of Φ sum to 269). The radii r_0^2 are $2 \cdot 10^{-4}$, $4 \cdot 10^{-6}$, $1.5 \cdot 10^{-5}$ and $6 \cdot 10^{-10}$ for $N = 1, \ldots, 4$: the detector fires only when the damper has brought the system almost exactly to rest. Because the funnel is entered at a single point, no wider coverage of the swing-up trajectory is needed.

C. Thresholding by a saturating amplifier

To charge the latch with a clean pulse we pass b through five ReLU stages of gain $g = 185$ and a clipped pair,

$$\beta = \text{ReLU}(g a_5) - \text{ReLU}(g a_5 - 1) = \min\big(1,\ g^6 b\big), \tag{9}$$

where $a_1 = \text{ReLU}(gb)$ and $a_k = \text{ReLU}(g a_{k-1})$. Since $g^6 \approx 4.0 \cdot 10^{13}$, any $b \geq 2.5 \cdot 10^{-14}$ yields $\beta = 1$. As b is obtained by cancelling terms of magnitude $\sim 10^2$ in double precision, its smallest positive values are of the same order as this threshold, so β is binary in nearly all cases; fractional values can occur only within a few units in the last place of the detector boundary. This is a deliberate use of floating-point resolution and is discussed in Section VII.

D. Latch and arbitration

The latch is a ReLU unit with a self-edge of weight +1:

$$L^t = \text{ReLU}\left(L^{t-1} + \beta^t - 1/T - 0.002\,g_\uparrow^t\right). \tag{10}$$

One detection charges L to 1, which then decays linearly to 0 in T steps, where T is the measured duration of the nominal swing-up from rest plus 250 steps (0.5 s). Inside the upright cone $g_\uparrow$ is large (≈ 94 when fully upright for $N = 4$) and resets L within a few steps, whereas brief cone crossings during the swing-up (where $g_\uparrow$ is small) do not.

A further gate $g_{\text{damp}} = \text{ReLU}(2 - 250L - 200 g_\uparrow)$ is subtracted from the swing-up routers, and the damper routers $\text{ReLU}(\pm u_{\text{damp}}/F - 250L - 200 g_\uparrow)$ are added to the output. The resulting modes are:

- *Balance* ($g_\uparrow > 0$): only the upright branch acts; L is reset.
- *Damp* (outside the cone, $L = 0$): the upright branch is inhibited by $g_\downarrow$, the swing-up branch by $g_{\text{damp}} = 2 > 1.25\,|u_{\text{sw}}|$; only u_{damp} acts.
- *Swing* ($L > 0$): $g_{\text{damp}} = 0$ and the damper is inhibited; the learned swing-up acts exactly as in the baseline.

TABLE II
NETWORK SIZES (HIDDEN NEURONS / ENABLED CONNECTIONS).

		Baseline (0.2)		With funnel (0.3)	
N	Inputs	Hidden	Conn.	Hidden	Conn.
1	5	9	37	24	73
2	8	13	80	29	123
3	11	16	125	33	175
4	14	20	183	38	240

The funnel adds $15 + N$ hidden neurons and between 36 and 57 connections; Table II gives the sizes. The largest new weight is 250, within the original bound of 300.

VI. EXPERIMENTS

A. Protocol

We compare, for each N, the bundled baseline network (repository tag 0.2) with the same network plus the funnel (tag 0.3). All episodes last 25 s. An episode is a *success* if $\min_i \cos\theta_i > 0.99$ holds continuously for at least the final 1 s; the *settle time* is the time after which it holds until the end. Contact with the compliant walls is recorded but is not a failure, since the walls are part of the plant. Scenarios:

- *Swing-up from rest*: $o = o_\downarrow$.
- *Random knock-downs* (200 per N, fixed seed): $x \sim \mathcal{U}[-2.1, 2.1]$ m, $\dot{x} \sim \mathcal{U}[-3.5, 3.5]$ m/s, $\theta_i \sim \mathcal{U}[-\pi, \pi)$, $\dot{\theta}_i \sim \mathcal{U}[-6, 6]$ rad/s.
- *Random upright tilts* (50 per N): $\theta_i \sim \mathcal{U}[-a, a]$ independently per link, at rest, with $a = 0.5, 0.22, 0.11, 0.035$ rad.
- *Extreme pushes*: from upright, a mouse leash (stiffness 60 N/m, damping 8 N s/m, force cap 250 N) pulls the cart or the tip toward a target at height -0.5 m and horizontal position ± 2.0 or ± 2.5 m for 0.3–0.9 s, starting at $t = 0.1$ s.

The harness is deterministic and is included in the repository (`paper/eval`); a full run takes about 6 s on a 14-core Apple M4 Pro.

B. Results

Nominal behavior is unchanged. Both controllers swing up from rest with identical settle times (0.9, 1.2, 2.7 and 4.0 s; Table III). At $o_\downarrow$ the detector fires immediately, the latch engages, and the learned swing-up runs exactly as before.

The baseline does not re-enter for $N \geq 2$. From random knock-down states the baseline recovers in 194/200 trials for $N = 1$, but in only 2, 1 and 0 of 200 trials for $N = 2, 3, 4$, and it fails all 18 extreme pushes for $N \geq 2$ (Tables III–IV). For $N = 1$, the learned policy behaves like an energy pump and generalizes; for longer chains it does not.

The funnel restores re-entry. With the funnel, all 800 knock-downs and all 24 pushes are recovered. The cost is time: the damper must bring the system almost exactly to rest before the swing-up starts, so the median settle time grows from 3.3 s ($N = 1$) to 12.6 s ($N = 4$), and the worst push for $N = 4$ takes 16.7 s. Wall contact also drops sharply (e.g. 200 to 87 trials for $N = 4$), but does not disappear; many knock-down states start near the track end with velocity toward the wall.

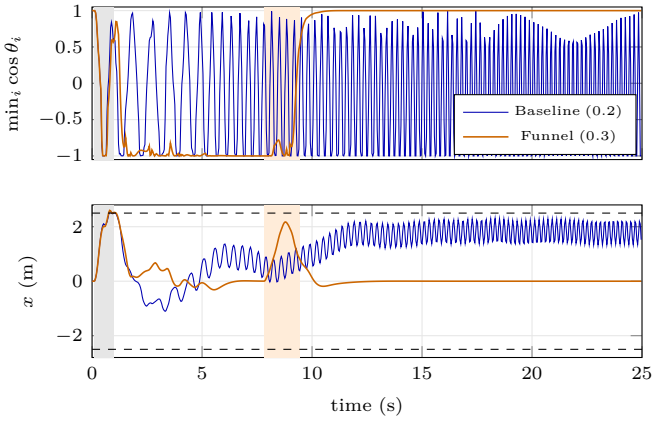


Fig. 2. $N = 3$, cart pushed toward +2.5 m for 0.9 s (gray band). Orange band: latch active (swing mode) for the funnel controller. Dashed: track limits.

The funnel does not enlarge the upright basin. For random per-link tilts the baseline's upright branch catches only 17/50 ($N = 3$) and 13/50 ($N = 4$) cases. The funnel controller succeeds in 50/50, but Table V shows that for $N = 3, 4$ most successes (33 and 37 trials) come from falling and re-swinging, with median settle times of 8.4 and 14.2 s. The funnel guarantees eventual re-entry; it does not improve the local stabilizer.

Fig. 2 shows a representative $N = 3$ episode. During and after a 0.9 s cart push the chain falls. The baseline keeps swinging and the cart repeatedly hits the wall. The funnel controller damps the chain and recenters the cart, the latch engages at $t \approx 7.8$ s, and the chain is upright at $t \approx 10.6$ s.

VII. LIMITATIONS

Noise-free assumption. The detector radii are tiny; for $N = 4$, $r_0^2 = 6 \cdot 10^{-10}$ requires $|\theta_i - \pi| \lesssim 5 \cdot 10^{-5}$ rad. With realistic sensor noise the detector would never fire and the controller would remain in damp mode. A practical version needs a larger, noise-aware entrance region and a swing-up that tolerates imperfect initial states.

Floating-point dependence. Binary thresholding relies on the resolution of double-precision arithmetic. Lower precision, a different summation order, or fused operations can change β near the boundary.

Designed, not learned. The funnel's structure and constants (r_0, T, gate weights, LQR weights) were chosen by hand per N and verified only empirically. We provide no proof that the damper's basin covers all reachable states or that T always suffices.

TABLE III

Swing-up from rest and recovery from 200 random knock-down states per link count. Times in seconds; settle-time statistics are over successful trials of the funnel controller.

	Swing-up from rest (settle time)		Knock-down recovery		Settle time (0.3)		Trials with wall contact	
N	Baseline (0.2)	Funnel (0.3)	Baseline (0.2)	Funnel (0.3)	median	90th pct.	Baseline (0.2)	Funnel (0.3)
1	✓ (0.9)	✓ (0.9)	194/200	200/200	3.3	3.7	81	6
2	✓ (1.2)	✓ (1.2)	2/200	200/200	5.9	7.2	198	13
3	✓ (2.7)	✓ (2.7)	1/200	200/200	7.6	9.0	193	40
4	✓ (4.0)	✓ (4.0)	0/200	200/200	12.6	14.8	200	87

TABLE IV

Extreme push scenarios from upright. Columns 4–9: settle time (s) of the funnel controller.

	Recovered		Cart push ±2.0 m, 0.3 s		Cart push ±2.5 m, 0.9 s		Tip yank ±2.0 m, 0.7 s	
N	Baseline (0.2)	Funnel (0.3)	+	−	+	−	+	−
1	6/6	6/6	1.9	1.9	2.3	2.3	4.0	4.0
2	0/6	6/6	7.8	7.0	7.7	8.6	6.1	7.0
3	0/6	6/6	10.9	10.9	10.6	10.6	7.3	7.4
4	0/6	6/6	13.3	13.3	16.7	16.7	11.8	11.8

TABLE V

Random upright tilts (50 per N). "Latched": trials in which the funnel controller fell and recovered through the full damp–swing cycle.

		Success			Median
N	a (rad)	0.2	0.3	Latched	settle (s)
1	0.500	50/50	50/50	0	0.1
2	0.220	45/50	50/50	5	0.4
3	0.110	17/50	50/50	33	8.4
4	0.035	13/50	50/50	37	14.2

Slow recovery. Waiting for full rest is conservative; median recovery for $N = 4$ exceeds 12 s. Entering the swing-up from a family of states, or learning a policy that re-enters directly, would be faster.

Scope. All results are in simulation with nominal parameters, no actuator dynamics, no delay, and $N \leq 4$. Wall contacts were not penalized.

VIII. Conclusion

A swing-up policy optimized from rest can be reliable from rest and still useless after a disturbance. For $N \geq 2$ links we found this to be nearly total: at most 1% recovery from random knock-downs. Adding a small, designed circuit (a downward LQR damper, a rest detector and a single self-recurrent latch neuron) to the trained network, without retraining, restored recovery to 100% in our tests and left nominal behavior unchanged. The main open problems are robustness to noise and faster re-entry; a natural next step is to use the funnel as a teacher or curriculum for learning a policy that re-enters without waiting for full rest.

Code Availability

The simulator, controllers, trained models, training data, and evaluation datasets used for every number in this paper are included in the open-source project repository.

CRediT Authorship Contribution Statement

Shashank: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data Curation, Visualization, Writing – Original Draft. **Mayank Goswami:** Supervision, Project administration, Technical guidance, Writing – Review & Editing.

IX. Video

https://youtu.be/TTnCf_FYlfU

References

[1] K. J. Åström and K. Furuta, "Swinging up a pendulum by energy control," *Automatica*, vol. 36, no. 2, pp. 287–295, 2000.

[2] M. W. Spong, "The swing up control problem for the Acrobot," *IEEE Control Systems Magazine*, vol. 15, no. 1, pp. 49–55, 1995.

[3] K. Graichen, M. Treuer, and M. Zeitz, "Swing-up of the double pendulum on a cart by feedforward and feedback control with experimental validation," *Automatica*, vol. 43, no. 1, pp. 63–71, 2007.

[4] T. Glück, A. Eder, and A. Kugi, "Swing-up control of a triple pendulum on a cart with experimental validation," *Automatica*, vol. 49, no. 3, pp. 801–808, 2013.

[5] A. G. Barto, R. S. Sutton, and C. W. Anderson, "Neuronlike adaptive elements that can solve difficult learning control problems," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-13, no. 5, pp. 834–846, 1983.

[6] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evolutionary Computation*, vol. 10, no. 2, pp. 99–127, 2002.

[7] F. Gomez, J. Schmidhuber, and R. Miikkulainen, "Accelerated neural evolution through cooperatively coevolved synapses," *Journal of Machine Learning Research*, vol. 9, pp. 937–965, 2008.

[8] R. R. Burridge, A. A. Rizzi, and D. E. Koditschek, "Sequential composition of dynamically dexterous robot behaviors," *The International Journal of Robotics Research*, vol. 18, no. 6, pp. 534–555, 1999.

[9] R. Tedrake, I. R. Manchester, M. Tobenkin, and J. W. Roberts, "LQR-trees: Feedback motion planning via sums-of-squares verification," *The International Journal of Robotics Research*, vol. 29, no. 8, pp. 1038–1052, 2010.

[10] B. D. O. Anderson and J. B. Moore, *Optimal Control: Linear Quadratic Methods*. Englewood Cliffs, NJ: Prentice Hall, 1990.
[11] L. M. Bregman, "The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming," *USSR Computational Mathematics and Mathematical Physics*, vol. 7, no. 3, pp. 200–217, 1967.